Interference Search

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	64	non-deterministic adj event	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:13
L2	781	(storage stor$3) near2 (checkpoint$3 check-point$3 "check point")	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:18
L4	275	(register) same processor same (checkpoint$3 check-point$3 "check point")	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:23
L5	1379040	(error$5 or fault$3 or problem or malfunction or fail$4) same (after subsequent follow$3 trail$3)	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:26
L6	161	2 same 5	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:26
L7	6	6 and 1	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:26
L8	40	6 and 4	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:26
L9	708	(714/2).ccls.	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:28
L10	647	(714/13).ccls.	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:28
L11	826	(714/15).ccls.	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:28
L12	1810	(714/38).ccls.	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:44
L13	244	(714/16).ccls.	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:44
L14	5317	hardware same (error$5 or fault$3 or problem or malfunction or fail$4) same recover$3	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:49
L15	0	8 and 9	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:49
L16	6	8 and 10	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:50

EAST Search History

L17	6	8 and 11	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:50
L18	3	8 and 12	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:50
L19	1	8 and 13	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:50
L20	3388	(lead$3 or trail$3) adj thread	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:51
L21	183	((lead$3 or primary) adj thread) same ((trail$3 or secondary or following) adj thread)	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:57
L22	73	21 and 5	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 15:57
L23	45	21 same 5	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 16:02
L24	2	23 and 9	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 16:02
L25	1	23 and 10	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 16:03
L26	2	23 and 11	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 16:03
L27	1	23 and 12	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 16:03
L28	0	23 and 13	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 16:04
L29	15	23 and ("714"/$).ccls.	US-PGPUB; USPAT; USOCR	OR	ON	2007/12/28 16:05
L30	3	29 and 1 and 2	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/12/28 16:05

Updated Search 10/651,523

Terms used:
hardware and **recovery** and **multi** **thread** and **checkpoint** and **error** and **subsequent** and **storage**

Found **23,632** of **216,199**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 ReVive: cost-effective architectural support for rollback recovery in shared-memory multiprocessors

Milos Prvulovic, Zheng Zhang, Josep Torrellas

May 2002 **ACM SIGARCH Computer Architecture News , Proceedings of the 29th annual international symposium on Computer architecture ISCA '02**, Volume 30 Issue 2

Publisher: IEEE Computer Society, ACM

Full text available: pdf(1.38 MB) Publisher Site

Additional Information: full citation, abstract, references, cited by, index terms

This paper presents ReVive, a novel general-purpose rollback recovery mechanism for shared-memory multiprocessors. ReVive carefully balances the conflicting requirements of availability, performance, and hardware cost. ReVive performs checkpointing, logging, and distributed parity protection, all memory-based. It enables recovery from a wide class of errors, including the permanent loss of an entire node. To maintain high performance, ReVive includes specialized hardware that performs frequent o ...

Keywords: fault tolerance, shared-memory multiprocessors, rollback recovery, recovery, BER, logging, parity, checkpointing, availability

2 SafetyNet: improving the availability of shared memory multiprocessors with global checkpoint/recovery

Daniel J. Sorin, Milo M. K. Martin, Mark D. Hill, David A. Wood

May 2002 **ACM SIGARCH Computer Architecture News , Proceedings of the 29th annual international symposium on Computer architecture ISCA '02**, Volume 30 Issue 2

Publisher: IEEE Computer Society, ACM

Full text available: pdf(1.28 MB) Publisher Site

Additional Information: full citation, abstract, references, cited by, index terms

We develop an availability solution, called *SafetyNet,* that uses a unified, lightweight checkpoint/recovery mechanism to support multiple long-latency fault detection schemes. At an abstract level, *SafetyNet* logically maintains multiple, globally consistent checkpoints of the state of a shared memory multiprocessor (i.e., processors, memory, and coherence permissions), and it recovers to a pre-fault checkpoint of the system and

Terms used:
hardware and **recovery** and **multi** **thread** and **checkpoint** and **error** and **subsequent** and **storage**

Found **23,632** of **216,199**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 81 - 100 of 200 Result page: previous 1 2 3 4 **5** 6 7 8 9 10 next

Best 200 shown

Relevance scale

81 Dynamic Verification of Sequential Consistency

Albert Meixner, Daniel J. Sorin

May 2005 **ACM SIGARCH Computer Architecture News , Proceedings of the 32nd annual international symposium on Computer Architecture ISCA '05**, Volume 33 Issue 2

Publisher: IEEE Computer Society, ACM Press

Full text available: pdf(146.68 KB) Additional Information: full citation, abstract, index terms

In this paper, we develop the first feasibly implementable scheme for end-to-end dynamic verification of multithreaded memory systems. For multithreaded (including multiprocessor) memory systems, end-to-end correctness is defined by its memory consistency model. One such consistency model is sequential consistency (SC), which specifies that all loads and stores appear to execute in a total order that respects program order for each thread. Our design, DVSC-Indirect, performs dynamic verification ...

82 An abstract model of rollback recovery control in distributed systems

Jiannong Cao, K. C. Wang

October 1992 **ACM SIGOPS Operating Systems Review**, Volume 26 Issue 4

Publisher: ACM Press

Full text available: pdf(1.30 MB) Additional Information: full citation, abstract, index terms

This paper develops an abstract model which presents a method of uniform description of different rollback recovery control algorithms for distributed systems. We first developed a general definition of the distributed rollback recovery control problem. The concept of a distributed recovery control system (*DRC system*), consisting of distributed recovery control units (*DRC units*), is proposed to model recovery with various control granularities. Then, we developed a graph model, cal ...

83 The space shuttle primary computer system

Alfred Spector, David Gifford

September 1984 **Communications of the ACM**, Volume 27 Issue 9

Publisher: ACM Press

Full text available: pdf(5.34 MB) Additional Information: full citation, references, citings, index terms

Keywords: PASS, avionics system, space shuttle

Updated Search 10/65/1523

Fri, 28 Dec 2007, 4:57:39 PM EST

Edit an existing query or compose a new query in the Search Query Display.

Search Query Display

Select a search number (#) to:

- Add a query to the Search Query Display
- Combine search queries using AND, OR, or NOT
- Delete a search
- Run a search

Recent Search Queries		Results
#1	(((hardware)<in>metadata) <and> ((multi-thread) <in>metadata))<and> ((recovery and checkpoint) <in>metadata)	0
#2	(((checkpoint)<in>metadata) <and> ((error occurred subsequent)<in>metadata))<and> ((storage)<in>metadata)	0
#3	(((register architectural state)<in>metadata) <and> ((checkpoint)<in>metadata))<and> ((error detect) <in>metadata)	0
#4	(((leading thread)<in>metadata) <and> ((trailing thread) <in>metadata))<and> ((subsequent checkpoint)<in>metadata)	0
#5	(((checkpoint)<in>metadata) <and> ((hardware recovery) <in>metadata))<and> ((multi-threaded)<in>metadata)	0
#6	(((hardware)<in>metadata) <and> ((recovery)<in>metadata)) <and> ((multi-threaded)<in>metadata)	7
#7	(((hardware)<in>metadata) <and> ((recovery)<in>metadata)) <and> ((multi-threaded)<in>metadata)	7
#8	(((hardware)<in>metadata) <and> ((recovery)<in>metadata)) <and> ((multi-threaded)<in>metadata)	7
#9	(((hardware)<in>metadata) <and> ((recovery)<in>metadata)) <and> ((multi-threaded)<in>metadata)	7
#10	(((hardware)<in>metadata) <and> ((recovery)<in>metadata)) <and> ((multi-threaded)<in>metadata)	7